

Minimum parametric flow over time

Mircea Parpalea¹

Nicoleta Avesalon²

Eleonor Ciurea²

¹ Andrei Șaguna National College, Brașov, Romania

² Transilvania University of Brașov, Romania

The paper presents a dynamic solution method for dynamic minimum parametric networks flow. The solution method solves the problem for a special parametric dynamic network with linear lower bound functions of a single parameter. Instead directly work on the original network, the method implements a labelling algorithm in the parametric dynamic residual network and uses quickest paths from the source node to the sink node in the time-space network along which repeatedly decreases the dynamic flow for a sequence of parameter values, in their increasing order. In each iteration, the algorithm computes both the minimum flow for a certain subinterval of the parameter values, and the new breakpoint for the maximum parametric dynamic flow value function.

Keywords: dynamic networks, parametric flow, partitioning algorithm

1 Introduction

Dynamic flow problems where networks structure changes depending on a scalar parameter λ are widely used (see for example Aronson (1989)) to model different network-structured, decision-making problems over time. These types of problems are arising in various real applications such as communication networks, air/road traffic control, and production systems. Moreover, in many applications of graph algorithms, including communication networks, graphics, assembly planning, and scheduling, graphs are subject to discrete changes, such as additions or deletions of arcs or nodes. In the last decade there has been a growing interest for such dynamically changing graphs, and a whole body of algorithms and data structures for dynamic graphs has been discovered: Fathabadi et al. (2015), He et al. (2015), Nassir et al. (2014), Rashidi and Tsang (2015) or Zheng et al. (2015). Further on, the next section presents some basic discrete-time dynamic networks terminology and notations and Section 3 introduces the parametric minimum flow over time problem. In Section 4 the algorithm for solving the parametric minimum flow in discrete dynamic networks is presented and Section 5 illustrates how the algorithm work on a given dynamic network.

2 Discrete-time dynamic network

A discrete dynamic network $G = (N, A, T)$ is a directed graph with N being a set of $|N| = n$ nodes i , A being a set of $|A| = m$ arcs a and T being the finite time horizon discretized into the set $H = \{0, 1, \dots, T\}$. An arc $a \in A$ from node i to node j is denoted by (i, j) . Parallel, as well as opposite arcs are not allowed in graph G . The following time-dependent functions are associated with each arc $a = (i, j) \in A$: the *upper bound (capacity)* function $u(i, j; \theta)$, $u : A \times H \rightarrow \mathbb{R}^+$, representing the maximum amount of flow that can enter the arc (i, j) at time θ , the *lower bound* function $\ell(i, j; \theta)$, $\ell : A \times H \rightarrow \mathbb{R}^+$, i.e. the minimum amount of flow that must enter the arc (i, j) at time θ , and the *transit time* function $h(i, j; \theta)$, $h : A \times H \rightarrow \mathbb{N}$. Time is measured in discrete steps, so that if one unit of flow leaves node i at time θ over the arc $a = (i, j)$ that one unit of flow arrives at node j at time $\theta + h(i, j; \theta)$, where $h(i, j; \theta)$ is the transit time of the arc (i, j) . The time horizon T represents the time limit until which the flow can travel in the network. The network has two special nodes: a source node s and a sink node t .

2.1 Time-space network

For a given discrete-time dynamic network, the *time-space network* is a static network constructed by expanding the original network in the time dimension, by considering a separate copy of every node $i \in N$ at every discrete time step $\theta \in H$. A *node-time pair* (NTP) (i, θ) refers to a particular node $i \in N$ at a particular time step $\theta \in H$, i.e., $(i, \theta) \in N \times H$.

Definition 1 The time-space network G^T of the original dynamic network G is defined as follows:

- (a) $N^T := \{(i, \theta) | i \in N, \theta \in H\}$;
- (b) $A^T := \{a_\theta = ((i, \theta), (j, \theta + h(i, j; \theta))) | 0 \leq \theta \leq T - h(i, j; \theta), (i, j) \in A\}$;
- (c) $u^T(a_\theta) := u(a; \theta)$ for $a_\theta \in A^T$;
- (d) $\ell^T(a_\theta) := \ell(a; \theta)$ for $a_\theta \in A^T$.

For every arc $(i, j) \in A$ with traversal time $h(i, j; \theta)$, capacity $u(i, j; \theta)$ and lower bound $\ell(i, j; \theta)$, the time-space network G^T contains the arcs $((i, \theta), (j, \theta + h(i, j; \theta)))$, $\theta = 0, 1, \dots, T - h(i, j; \theta)$ with capacities $u(i, j; \theta)$ and lower bounds $\ell(i, j; \theta)$. A (discrete-time) *dynamic path* \bar{P} is defined as a sequence of distinct, consecutively linked NTPs.

2.2 Parametric dynamic network

Definition 2 A discrete-time dynamic network $G = (N, A, T)$ for which the lower bounds $\ell(i, j; \theta)$ of some arcs $(i, j) \in A$ are functions of a real parameter λ is referred to as a *parametric dynamic network* and is denoted by $\bar{G} = (N, A, u, \bar{\ell}, h, T)$.

For a parametric dynamic network \bar{G} , the parametric lower bound function $\bar{\ell} : A \times H \times [0, \Lambda] \rightarrow \mathbb{R}^+$ associates to each arc $(i, j) \in A$ and for each of the parameter values λ in an interval $[0, \Lambda]$ the real number $\bar{\ell}(i, j; \theta; \lambda)$, referred to as the *parametric lower bound* of arc (i, j) :

$$\bar{\ell}(i, j; \theta; \lambda) = \ell_0(i, j; \theta) + \lambda \cdot \mathcal{L}(i, j; \theta), \quad \lambda \in [0, \Lambda], \quad \theta \in H, \quad (1)$$

where $\mathcal{L} : A \times H \rightarrow \mathbb{R}$ is a real valued function, associating to each arc $(i, j) \in A$ and for each time-value $\theta \in H$ the real number $\mathcal{L}(i, j; \theta)$, referred to as the *parametric part of the lower bound* of the arc (i, j) . The nonnegative value $\ell_0(i, j; \theta)$ is the lower bound of the arc (i, j) for $\lambda = 0$, i.e.,

$\bar{\ell}(i, j; \theta; 0) = \ell_0(i, j; \theta)$. For the problem to be correctly formulated, the lower bound function of every arc $(i, j) \in A$ must respect the condition $u(i, j; \theta) \geq \bar{\ell}(i, j; \theta; \lambda)$ for the entire interval of the parameter values, i.e., $\forall (i, j) \in A, \forall \theta \in H$ and $\forall \lambda \in [0, \Lambda]$. It follows that $u(i, j; \theta) \geq \ell_0(i, j; \theta)$ and the parametric part of the lower bounds $\mathcal{L}(i, j; \theta)$ must satisfy the constraints: $\mathcal{L}(i, j; \theta) \leq [u(i, j; \theta) - \ell_0(i, j; \theta)]/\Lambda$, $\forall (i, j) \in A$.

3 Parametric flow over time

The *parametric dynamic flow value function* $\bar{v} : N \times H \times [0, \Lambda] \rightarrow \mathbb{R}$ associates to each of the nodes $i \in N$, at each time moment $\theta \in H$, a real number $\bar{v}(i; \theta; \lambda)$ referred to as the value of node i at time θ , for each of the parameter λ values.

Definition 3 A feasible parametric flow over time $\bar{f}(i, j; \theta; \lambda)$ in a parametric dynamic network $\bar{G} = (N, A, u, \bar{\ell}, h, T)$ is a function $\bar{f} : A \times H \times [0, \Lambda] \rightarrow \mathbb{R}^+$ that satisfies the following constraints for all $\lambda \in [0, \Lambda]$:

$$\sum_{j|(i,j) \in A} \bar{f}(i, j; \theta; \lambda) - \sum_{j|(j,i) \in A} \sum_{\vartheta | \vartheta + h(j,i;\vartheta) = \theta} \bar{f}(j, i; \vartheta; \lambda) = \begin{cases} \bar{v}(i; \theta; \lambda), & i = s, t \\ 0, & i \neq s, t \end{cases}, \forall i \in N, \forall \theta \in H; \quad (2)$$

$$\bar{\ell}(i, j; \theta; \lambda) \leq \bar{f}(i, j; \theta; \lambda) \leq u(i, j; \theta), \quad \forall (i, j) \in A, \quad \forall \theta \in H; \quad (3)$$

$$\bar{f}(i, j; \theta; \lambda) = 0, \quad \forall (i, j) \in A, \quad \forall \theta \in [T - h(i, j; \theta) + 1, T]; \quad (4)$$

$$\sum_{\theta \in H} \bar{v}(s; \theta; \lambda) = - \sum_{\theta \in H} \bar{v}(t; \theta; \lambda) = \bar{v}(\lambda); \quad (5)$$

where $\bar{f}(i, j; \theta; \lambda)$ determines the rate of flow (per time unit) entering arc (i, j) at time θ , for the parameter value λ , $\forall \theta \in \{0, 1, \dots, T\}$ and $\forall \lambda \in [0, \Lambda]$.

Definition 4 The parametric minimum flow over time (PmFT) problem is to compute all minimum flows over time for every possible value of λ :

$$\text{minimise } \bar{v}(\lambda) = \sum_{\theta \in \{0, 1, \dots, T\}} \bar{v}(s; \theta; \lambda), \quad \text{for all } \lambda \in [0, \Lambda], \quad (6)$$

under flow constraints (2)-(5).

Definition 5 For the minimum flow over time problem, the parametric time-dependent residual network with respect to a given feasible parametric flow over time \bar{f} is defined as $\bar{G}(\bar{f}) := (N, A(\bar{f}), T)$, with $A(\bar{f}) := A^+(\bar{f}) \cup A^-(\bar{f})$, where

$$A^+(\bar{f}) := \{(i, j) | (i, j) \in A, \exists \theta \leq T - h(i, j; \theta) \text{ with } \bar{f}(i, j; \theta; \lambda) - \bar{\ell}(i, j; \theta; \lambda) > 0\}; \quad (7)$$

$$A^-(\bar{f}) := \{(i, j) | (j, i) \in A, \exists \theta \leq T - h(j, i; \theta) \text{ with } u(j, i; \theta) - \bar{f}(j, i; \theta; \lambda) > 0\}. \quad (8)$$

The direct arcs $(i, j) \in A^+(\bar{f})$ in $\bar{G}(\bar{f})$ have same transit times $h(i, j; \theta)$ with those in \bar{G} while the reverse arcs $(i, j) \in A^-(\bar{f})$ have negative transit times $h(i, j; \theta + h(j, i; \theta)) = -h(j, i; \theta)$, with $(j, i) \in A$ and $0 \leq \theta + h(j, i; \theta) \leq T$.

Definition 6 For the parametric minimum flow over time (PmFT) problem, the parametric residual capacities of arcs (i, j) in the parametric time-dependent residual network $\bar{G}(\bar{f})$ are defined as follows:

$$\bar{r}(i, j; \theta; \lambda) = \bar{f}(i, j; \theta; \lambda) - \bar{\ell}(i, j; \theta; \lambda), \quad (i, j) \in A, \quad 0 \leq \theta + h(i, j; \theta) \leq T; \quad (9)$$

$$\bar{r}(i, j; \theta + h(j, i; \theta); \lambda) = u(j, i; \theta) - \bar{f}(j, i; \theta; \lambda), \quad (j, i) \in A, \quad 0 \leq \theta + h(j, i; \theta) \leq T. \quad (10)$$

Definition 7 Given a parametric flow over time $\bar{f}(i, j; \theta; \lambda)$, the parametric residual capacity $\bar{r}(\bar{P}; \theta; \lambda)$ of a dynamic path \bar{P} is the inner envelope of the parametric residual capacity functions $\bar{r}(i, j; \theta; \lambda)$ for all arcs (i, j) composing the path and for all parameter λ values:

$$\bar{r}(\bar{P}; \theta; \lambda) = \min\{\bar{r}(i, j; \theta; \lambda) | (i, j) \in \bar{P}\}. \quad (11)$$

Definition 8 The transit time $\tau(\bar{P}; \theta)$ of a dynamic path \bar{P} is defined by:

$$\tau(\bar{P}) = \sum_{(i, j) \in \bar{P}} h(i, j; \theta). \quad (12)$$

A dynamic path \bar{P} is referred to as the *quickest dynamic path* if $\tau(\bar{P}) \leq \tau(\bar{P}')$ for all dynamic paths \bar{P}' in the parametric time-dependent residual network $\bar{G}(\bar{f})$.

4 Parametric minimum dynamic flow

The approaches for solving the maximum parametric flow over time problem via applying classical algorithms can be grouped in two main categories: by applying a classical parametric flow algorithm (see Hamacher and Foulds (1989) and Ruhe (1991)) in the static time-space network (for this approach see the algorithm presented in Parpalea and Ciurea (2013)) or by applying a non-parametric maximum dynamic flow algorithm (see the algorithms presented in Parpalea and Ciurea (2011)) in dynamic residual networks generated by partitioning the interval of the parameter values (partitioning method presented by Avesalon et al.). Given the powerful versatility of dynamic algorithms, it is not surprising that these algorithms and dynamic data structures are often more difficult to design and analyse than their static counterparts (see Orlin (2013)). Cai et al. (2007) proved that the complexity of finding a shortest dynamic flow augmenting path, by exploring the forward and reverse arcs successively, is $O(nmT^2)$. For algorithms which explores the two sub-networks simultaneously, Miller-Hooks and Patterson (2004) also reported a complexity of $O(n^2T^2)$. By using special node addition and selection procedures, Nasrabadi and Hashemia (2010) succeeded to reduce significantly the number of node time pair that needs to be visited. The worst-case complexity of their algorithm is $O(nT(n + T))$. As far as we know, the problem of minimum flow over time in parametric networks has not been treated yet.

4.1 Parametric minimum dynamic flow (PmDF) algorithm

The idea of the algorithm is that if the parametric residual capacities for all arcs in $\bar{G}(\bar{f})$ are maintained linear functions of λ , with no break points, the problem can be solved via a slightly modified non-parametric

algorithm. Firstly, if one exists, a feasible flow must be established. The most convenient is this to be done in the nonparametric network $G' = (N, A, u, \ell', h, T)$ obtained from the initial network by replacing the parametric lower bound functions with the non-parametric ones: $\ell'(i, j; \theta) = \max\{\ell(i, j; \theta; \lambda) | \lambda \in [0, \Lambda]\}$, i.e. $\ell'(i, j; \theta) = \ell_0(i, j; \theta)$ for $\mathcal{L}(i, j; \theta) \leq 0$ and $\ell'(i, j; \theta) = \ell_0(i, j; \theta) + \Lambda \cdot \mathcal{L}(i, j; \theta)$ for $\mathcal{L}(i, j; \theta) > 0$. For finding a feasible flow $\bar{f}(i, j; \theta)$ in \bar{G} see the algorithms presented in Ahuja et al. (1993), applied in the static time-space network. During the running of the algorithm, the subinterval of the parameter values is continuously narrowed so that the above restriction to remain valid. In each subinterval $[\lambda_k, \lambda_{k+1}]$ of the parameter values, the linear parametric residual capacity of every arc (i, j) can be written as $\bar{r}_k(i, j; \theta; \lambda) = \alpha_k(i, j; \theta) + \beta_k(i, j; \theta) \cdot (\lambda - \lambda_k)$ while the parametric flow is $\bar{f}_k(i, j; \theta; \lambda) = f_k(i, j; \theta) + F_k(i, j; \theta) \cdot (\lambda - \lambda_k)$. As soon as the parametric time-dependent residual network $\bar{G}(\bar{f})$ contains no dynamic paths, the algorithm computes the minimum flow for the considered subinterval and then reiterates on the next subinterval of the parameter values, until Λ value is reached.

```

(01) PmDF ALGORITHM;
(02) BEGIN
(03)   find a feasible flow  $\bar{f}(i, j; \theta)$  in network  $\bar{G}$ ;
(04)    $BP := \{0\}; \quad k := 0; \quad \lambda_k := 0;$ 
(05)   REPEAT
(06)     QDP( $k, \lambda_k, BP$ );
(07)      $k := k + 1;$ 
(08)   UNTIL( $\lambda_k = \Lambda$ );
(09) END.
```

Algorithm 1: Parametric minimum dynamic flow (PmDF) algorithm.

4.2 Quickest Dynamic Paths (QDP) procedure

The successive shortest (quickest) dynamic paths procedure repeatedly performs the following operations:

- Finds a shortest (quickest) dynamic path \bar{P} in the time-dependent residual network $\bar{G}(\bar{f})$ based on the successor vector $\sigma : N \times H \rightarrow (i, \theta) \in N \times H$;
- Computes the parametric residual capacity $\bar{r}(\bar{P}; \theta; \lambda)$ of the dynamic path \bar{P} ;
- Computes the first (in increasing order) value of the parameter λ up to which the parametric residual capacity of the dynamic path remains linear without break points;
- Updates the subinterval of the parameter values for which the minimum flow is determined;
- Decreases the flow along the dynamic path and updates the time-dependent residual network.

The algorithm ends when none of the source nodes, at any time moments $\theta \in \{0, 1, \dots, T\}$, is reachable from any of the sink nodes, i.e. there is no dynamic path from s to t .

```

(01) PROCEDURE QDP( $k, \lambda_k, BP$ );
(02) BEGIN
(03)   FOR all  $\theta \in H$  DO
(04)     BEGIN
(05)       FOR all  $i \in N$  DO  $\sigma(i, \theta) := (0, 0);$ 
(06)       FOR all  $(i, j) \in A$  DO
          $\alpha_k(i, j; \theta) := \bar{f}(i, j; \theta) - \ell_0(i, j; \theta) - \lambda_k \cdot \mathcal{L}(i, j; \theta); \beta_k(i, j; \theta) := -\mathcal{L}(i, j; \theta);$ 
          $\alpha_k(j, i; \theta + h(i, j; \theta)) := u(i, j; \theta) - \bar{f}(i, j; \theta); \beta_k(j, i; \theta + h(i, j; \theta)) := 0;$ 
          $f_k(i, j; \theta) := \bar{f}(i, j; \theta); F_k(i, j; \theta) := 0;$ 
(07)     END;
(08)    $C := 1; \lambda_{k+1} = \Lambda;$ 
(09)   LS( $\sigma, C$ );
```

```

(10) WHILE( $C = 1$ ) DO
(11)   BEGIN
(12)     build  $\bar{P}$  based on  $\sigma$  starting from  $(s, \bar{\theta})$ ;
(13)      $\alpha := \min\{\alpha_k(i, j; \theta) | (i, j) \in \bar{P}\}$ ;  $\beta := \min\{\beta_k(i, j; \theta) | (i, j) \in \bar{P}, \alpha_k(i, j; \theta) = \alpha\}$ ;
(14)      $(i, \theta) := (s, \bar{\theta})$ ;
(15)     WHILE( $i \neq t$ ) DO
(16)       BEGIN
(17)          $(j, \vartheta) := \sigma(i, \theta)$ ;
(18)         IF( $\beta_k(i, j; \theta) < \beta$ ) THEN  $\lambda_{k+1} := \min\{\lambda_{k+1}, \lambda_k + (\alpha_k(i, j; \theta) - \alpha) / (\beta - \beta_k(i, j; \theta))\}$ ;
(19)          $\alpha_k(i, j; \theta) := \alpha_k(i, j; \theta) - \alpha$ ;  $\beta_k(i, j; \theta) := \beta_k(i, j; \theta) - \beta$ ;
(20)          $\alpha_k(j, i; \vartheta) := \alpha_k(j, i; \vartheta) + \alpha$ ;
(21)          $\beta_k(j, i; \vartheta) := \beta_k(j, i; \vartheta) + \beta$ ;
(22)         IF( $(i, j) \in A^+(\bar{f})$ ) THEN  $f_k(i, j; \theta) := f_k(i, j; \theta) - \alpha$ ;  $F_k(i, j; \theta) := F_k(i, j; \theta) - \beta$ ;
(23)         ELSE  $f_k(j, i; \vartheta) := f_k(j, i; \vartheta) + \alpha$ ;  $F_k(j, i; \vartheta) := F_k(j, i; \vartheta) + \beta$ ;
(24)          $(i, \theta) := (j, \vartheta)$ ;
(25)       END;
(26)     FOR all  $\theta \in H$  DO FOR all  $i \in N$  DO  $\sigma(i, \theta) := (0, 0)$ ;
(27)     LS( $\sigma, C$ );
(28)   END.

```

Algorithm 2: Quickest Dynamic Paths (QDP) procedure.

4.3 Labels Setting (LS) procedure

The label setting procedure uses *transit time labels* $\tau(i, \theta)$ associated to all nodes at each discrete time values.

At any step of the algorithm, a label is *permanent* once it denotes the length of shortest augmenting path to a node-time pair, otherwise it is *temporary*. The *LS* procedure maintains a set L of candidate nodes in increasing order of their temporary labels, which initially includes only the sink nodes (t, θ) , $\theta \in \{0, 1, \dots, T\}$. For every node-time pair (j, ϑ) selected from the list, the arcs with positive residual capacity connecting (i, θ) to (j, ϑ) are explored, where $\vartheta = \theta + h(i, j; \theta)$ if the arc connecting (i, θ) to (j, ϑ) is a forward arc and $0 \leq \theta = \vartheta + h(j, i; \vartheta)$ if (i, j) is a reverse arc. At any iteration, the algorithm selects the node-time pair (i, θ) with the minimum temporary label, makes its transit time label permanent, checks optimality conditions and updates the labels accordingly. A transit time label $\tau(i, \theta)$ represents the length (transit time) of a shortest (quickest) dynamic path from (i, θ) if $\tau(i, \theta) \leq \tau(j, \vartheta) + h(j, i; \vartheta)$, $\forall (i, j) \in A(\bar{f})$. The process is repeated until there are no more candidate nodes in L . The transit time of the shortest (quickest) path \bar{P} computed based on successor vector σ is given by $\bar{\tau}$.

```

(01) PROCEDURE LS( $\sigma, C$ );
(02) BEGIN
(03)   FOR all  $\theta \in \{0, 1, \dots, T\}$  DO
(04)     BEGIN
(05)       FOR all  $i \in N - t$  DO  $\tau(i, \theta) := \infty$ ;
(06)        $\tau(t, \theta) := 0$ ;  $L := L \cup \{(t, \theta)\}$ ;
(07)     END;
(08)    $\bar{\tau} := \infty$ ;  $\bar{\theta} := \infty$ 
(09)   WHILE( $L \neq \emptyset$ ) DO
(10)     BEGIN
(11)       select the first  $(j, \vartheta)$  from  $L$ ;  $L := L - \{(j, \vartheta)\}$ ;
(12)       FOR all  $i \in N$  with  $(i, j) \in A^+(\bar{f})$  DO

```

```

(13)   FOR all  $\theta$  with  $\theta + h(i, j; \theta) = \vartheta$  DO
(14)   IF( $\tau(j, \vartheta) + h(i, j; \theta) < \tau(i, \theta)$ ) THEN
(15)   BEGIN
(16)    $\tau(i, \theta) := \tau(j, \vartheta) + h(i, j; \theta)$ 
(17)    $\sigma(i, \theta) := (j, \vartheta)$ 
(18)   IF( $(i, \theta) \notin L$ ) THEN  $L := L \cup \{(i, \theta)\}$ ;
(19)   END;
(20)   FOR all  $i \in N$  with  $(i, j) \in A^-(\bar{f})$  DO
(21)   IF( $\vartheta + h(j, i; \vartheta) \leq T$  and  $\tau(j, \vartheta) - h(j, i; \vartheta) < \tau(i, \vartheta + h(j, i; \vartheta))$ ) THEN
(22)   BEGIN
(23)    $\tau(i, \vartheta + h(j, i; \vartheta)) := \tau(j, \vartheta) - h(j, i; \vartheta)$ ;
(24)    $\sigma(i, \vartheta + h(j, i; \vartheta)) := (j, \vartheta)$ ;
(25)   IF( $(i, \vartheta + h(j, i; \vartheta)) \notin L$ ) THEN  $L := L \cup \{(i, \vartheta + h(j, i; \vartheta))\}$ ;
(26)   END;
(27)   END;
(28)    $\bar{\tau} := \min_{\theta \in \{0, 1, \dots, T\}} \{\tau(s, \theta)\}$ ;  $\bar{\theta} := \min_{\theta \in \{0, 1, \dots, T\}} \{\theta | \tau(s, \theta) = \bar{\tau}\}$ ;
(29)   IF( $\bar{\tau} = \infty$ ) THEN  $C := 0$ ;
(30) END.

```

Algorithm 3: Labels Setting (LS) procedure.

Theorem 1 [Correctness of PmDF algorithm] *PmDF algorithm computes correctly a parametric minimum flow over time for a given time horizon T and for $\lambda \in [0, \Lambda]$.*

Proof: The partitioning type algorithm iterates on successive subintervals $[\lambda_k, \lambda_{k+1}]$, starting with $\lambda_0 = 0$ and ending with $\lambda_{k_{max}+1} = \Lambda$ and consequently, the correctness of the algorithm obviously follows from the correctness of the Quickest Dynamic Paths (QDP) procedure which ends when none of the source node-time pairs is reachable from any of the sink node-time pairs, i.e. when there is no dynamic path from the source node to the sink node in the time-dependent residual network. According to the classical flow decreasing path theorem presented by Ahuja et al. (1993), this means that the obtained flow is a minimum dynamic flow for the given time horizon. In fact, the algorithm ends with a set of linear parametric minimum flows and with the partition BP of the interval of the parameter values in their corresponding subintervals. \square

Theorem 2 [Time complexity of PmDF algorithm] *The parametric minimum dynamic flow (PmDF) algorithm runs in $O(Kn^2mT^3)$ time, where $K + 1$ is the number of λ values in the set BP at the end of the algorithm.*

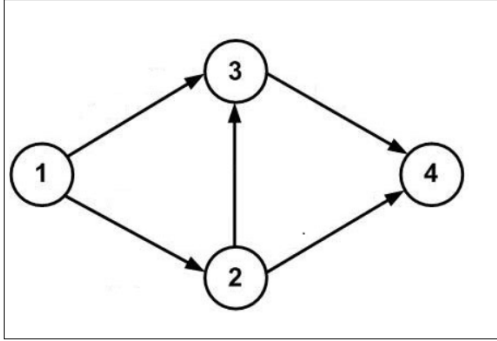
Proof: The building, as well as the updating of the time-dependent residual network requires an $O(mT)$ running time, since for each of the time values $0 \leq \theta \leq T$ all the m arcs must be examined. Labels Setting (LS) procedure investigates at most nT adjacent node-time pairs for each of the node-time pairs which are removed from the list L (i.e. $O(nT)$ times). Thus, the complexity of labels setting (LS) procedure is $O(n^2T^2)$. Considering that in each of the iterations of the QDP procedure, for one of the time values, one arc is eliminated from the dynamic residual network, the algorithm ends in at most $O(mT)$ iterations. On each of the iterations the procedure finds a quickest dynamic path with the complexity $O(n^2T^2)$ and updates the time-dependent residual network in $O(mT)$ time. Thus, the total complexity of the quickest dynamic paths (QDP) procedure is $O(n^2mT^3)$.

For each of the K subintervals $[\lambda_k, \lambda_{k+1}]$, $k = 0, 1, \dots, K-1$ in which the interval $[0, \Lambda]$ of the parameter

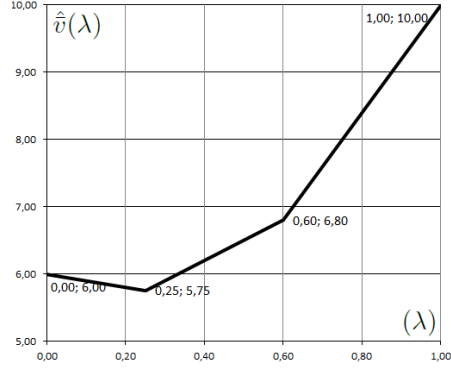
λ values is partitioned in, the algorithm makes a call to procedure QDP. Consequently, the complexity of the parametric minimum dynamic flow (PmDF) algorithm is $O(Kn^2mT^3)$. \square

5 Example

In the discrete-time dynamic network presented in Figure 1(a), node 1 is the source node s and node 4 is the sink node t ; the time horizon being set to $T = 3$, i.e. $H = \{0, 1, 2, 3\}$.



(a) The discrete-time dynamic network \bar{G} .



(b) The piecewise linear minimum flow over time value function for \bar{G} .

Fig. 1: Illustrating PmDF algorithm.

For the interval of the parameter λ values, set to $[0, 1]$, i.e. $\Lambda = 1$, the transit times $h(i, j; \theta)$, parametric lower bound functions $\bar{\ell}(i, j; \theta; \lambda) = \ell_0(i, j; \theta) + \lambda \cdot \mathcal{L}(i, j; \theta)$, upper bounds (capacities) $u(i, j; \theta)$ and feasible flow $\bar{f}(i, j; \theta)$ for all arcs in \bar{G} are indicated in Table 1.

Tab. 1: Characteristics of dynamic network \bar{G} presented in Figure 1(a)

(i, j)	$h(i, j; \theta)$	$\ell_0(i, j; \theta)$	$\mathcal{L}(i, j; \theta)$	$u(i, j; \theta)$	$\bar{f}(i, j; \theta)$
(1, 2)	1, $\theta = 0$	3, $\theta = 0$	-2, $\theta = 0$	5	5, $\theta = 0$
	2, $\theta \geq 1$	0, $\theta \geq 1$	0, $\theta \geq 1$		0, $\theta \geq 1$
(1, 3)	1, $0 \leq \theta < 2$	1, $0 \leq \theta < 2$	4, $\theta = 0$	5	5, $\theta = 0$
	2, $\theta \geq 2$	0, $\theta \geq 2$	1, $\theta = 1$		2, $\theta = 1$
(2, 3)	1, $\theta \geq 0$	0, $\theta \geq 0$	0, $\theta > 1$	5	0, $\theta \geq 1$
			3, $\theta = 1$		3, $\theta = 1$
(2, 4)	1, $0 \leq \theta < 2$	0, $\theta \geq 0$	0, $\theta \geq 0$	5	0, $\theta \neq 1$
	2, $\theta \geq 2$				2, $\theta = 1$
(3, 4)	2, $0 \leq \theta < 2$	0, $\theta = 0$	-2, $\theta = 2$	5	0, $\theta = 0$
	1, $\theta \geq 2$	2, $0 < \theta \leq 2$	0, $\theta \neq 2$		5, $0 < \theta \leq 2$
		0, $\theta > 2$			0, $\theta > 2$

After the initialisation step, for $k = 0$ and for the corresponding initial value of the parameter $\lambda_0 = 0$, procedure QDP is called for the first time. The successor vector is initialised for all nodes at all time values to $\sigma(i, \theta) := (0, 0)$ and the Labels setting (LS) procedure is called for finding a quickest dynamic path in the time-dependent residual network $\bar{G}(\bar{f})$. The distance labels are initialised to $\tau(4, \theta) := 0, \forall \theta \in \{0, 1, 2, 3\}$ and the set of candidate nodes is set to $L := \{(4, 0), (4, 1), (4, 2), (4, 3)\}$. After setting transit time labels and finding successor values for all node-time pairs, the procedure computes the minimum label of the source node, $\bar{\tau} := \min\{\tau(1, 0), \tau(1, 1), \tau(1, 2), \tau(1, 3)\} = \min\{2, 1, \infty, \infty\} = 1$ with $\bar{\theta} := 1$. Since $\bar{\tau}(4) \neq \infty$ the procedure LS ends with the variable C keeping its initial value $C = 1$. Based on successor vector, the quickest dynamic path $\bar{P} := ((1, 1), (3, 2), (2, 1), (4, 2))$ is built and its residual capacity $\bar{r}(\bar{P}; \theta; \lambda) = \alpha + (\lambda - \lambda_k) \cdot \beta$ is computed with $\alpha = \min\{\alpha_0(1, 3; 1), \alpha_0(3, 2; 2), \alpha_0(2, 4; 1)\} = \min\{1, 2, 2\} = 1$ and $\beta = -1$. After the time-dependent residual network is updated and the parametric dynamic flow is decreased along the shortest dynamic path, the successor vector is reinitialised and procedure LS is called again. The next shortest dynamic path found by QDP procedure is $\bar{P} := ((1, 0), (2, 1), (4, 2))$ with $\bar{r}(\bar{P}; \theta; \lambda) = 1 + \lambda$ and both the time-dependent residual network and the parametric dynamic flow are accordingly updated. Then QDP procedure finds the new dynamic path $\bar{P} := ((1, 0), (3, 1), (4, 3))$ with $\alpha = 3$ and $\beta = 0$. Since $\beta_0(1, 3; 0) = -4 < \beta = 0$, the upper limit λ_{k+1} of the subinterval of the parameter is updated to $\lambda_1 = \min\{1, (4 - 3)/(0 + 4)\} = 1/4$. Finally, after updating the time-dependent residual network and the parametric dynamic flow, based on successor vector found by LS procedure, QDP finds the last path $\bar{P} := ((1, 0), (2, 1), (3, 2), (4, 3))$ with $\alpha = 1$ and $\beta = 1$. The validity of the upper limit of the subinterval of the parameter values is tested and it is maintained unchanged since for the arc $(2, 3)$ at $\theta = 1$, $\beta_0(2, 3; 1) = -4 < \beta = 1$ but $\lambda_1 = \min\{1/4, (4 - 1)/(1 + 4)\} = \min\{1/4, 3/5\} = 1/4$. At this point, after the updating step, no other dynamic path can be found in the time-dependent residual network so that $C := 0$ is set, the breakpoints list is updated to $BP = \{0, 1/4\}$ and the first iteration ends with the minimum parametric flow over time $\hat{f}_0(i, j; \theta; \lambda)$ computed for the subinterval $[0, 1/4]$ of the parameter λ values. The PmDF algorithm increments variable k to the value $k = 1$ and a next iteration is performed. The evolution of the algorithm is presented in Table 2. The piecewise linear minimum flow over time value function for the discrete dynamic network \bar{G} , computed by *Parametric minimum Dynamic Flow* (PmDF) algorithm, is presented in Figure 1(b).

Tab. 2: The evolution of PmDF algorithm for the dynamic network \bar{G}

	k	λ_k	\bar{P}	$\bar{r}_k(\bar{P}; \theta; \lambda)$	λ_{k+1}
Iteration 1 :	0	0	$(1, 1), (3, 2), (2, 1), (4, 2)$	$1 - \lambda$	1
			$(1, 0), (2, 1), (4, 2)$	$1 + \lambda$	1
			$(1, 0), (3, 1), (4, 3)$	3	1/4
			$(1, 0), (2, 1), (3, 2), (4, 3)$	$1 + \lambda$	1/4
Iteration 2 :	1	1/4	$(1, 1), (3, 2), (2, 1), (4, 2)$	$1 - \lambda$	1
			$(1, 0), (2, 1), (4, 2)$	$1 + \lambda$	1
			$(1, 0), (3, 1), (4, 3)$	$4 - 4\lambda$	1
			$(1, 0), (2, 1), (3, 2), (4, 3)$	$1 + \lambda$	3/5
Iteration 3 :	2	3/5	$(1, 1), (3, 2), (2, 1), (4, 2)$	$1 - \lambda$	1
			$(1, 0), (2, 1), (4, 2)$	$1 + \lambda$	1
			$(1, 0), (3, 1), (4, 3)$	$4 - 4\lambda$	1
			$(1, 0), (2, 1), (3, 2), (4, 3)$	$4 - 4\lambda$	1

References

R. Ahuja, T. Magnanti, and J. Orlin. *Network Flows. Theory, algorithms and applications*. Prentice Hall Inc., Englewood Cliffs, New Jersey, 1993.

- J. Aronson. A survey of dynamic network flows. *Annals of Operations Research*, 20(5):1–66, 1989.
- N. Avesalon, E. Ciurea, and M. Parpalea. Maximum parametric flow in discrete-time dynamic networks. *Fundamenta Informaticae*, (accepted for publication).
- X. Cai, D. Sha, and C. Wong. *Time-varying Network Optimization*. Springer, New York, 2007.
- H. S. Fathabadi, S. Khezri, and S. Khodayifar. A simple algorithm for reliability evaluation in dynamic networks with stochastic transit times. *Journal of Industrial and Production Engineering*, 32(2):115–120, 2015. doi: 10.1080/21681015.2015.1015460.
- H.-W. Hamacher and L. Foulds. Algorithms for flows with parametric capacities. *Zeitschrift für Operations Research - Methods and Models of Operations Research*, 33(1):21–37, 1989.
- X. He, H. Zheng, and S. Peeta. Model and a solution algorithm for the dynamic resource allocation problem for large-scale transportation network evacuation. *Transportation Research Part C: Emerging Technologies*, 7:441–458, 2015. doi: 10.1016/j.trpro.2015.06.023.
- E. Miller-Hooks and S. S. Patterson. On solving quickest time problems in time-dependent, dynamic networks. *Journal of Mathematical Modelling and Algorithms*, 3(1):39–71, 2004.
- E. Nasrabadi and M. S. Hashemia. Minimum cost time-varying network flow problems. *Optimization Methods and Software*, 25(3):429–447, 2010. doi: 10.1080/10556780903239121.
- N. Nassir, H. Zheng, and M. Hickman. Efficient negative cycle-canceling algorithm for finding the optimal traffic routing for network evacuation with nonuniform threats. *Transportation Research Record: Journal of the Transportation Research Board*, 2459:81–90, 2014. doi: <http://dx.doi.org/10.3141/2459-10>.
- J. B. Orlin. Max flows in $o(nm)$ time, or better. In *Proceedings of the forty-fifth annual ACM symposium on Theory of computing*, pages 765–774, New York, 2013. ACM Press. doi: 10.1145/2488608.2488705.
- M. Parpalea and E. Ciurea. The quickest maximum dynamic flow of minimum cost. *International Journal of Applied Mathematics and Informatics*, 3(5):266–274, 2011.
- M. Parpalea and E. Ciurea. Partitioning algorithm for the parametric maximum flow. *Applied Mathematics*, 4(10A): 3–10, 2013. doi: 10.4236/am.2013.410A1002.
- H. Rashidi and E. Tsang. *Vehicle Scheduling in Port Automation: Advanced Algorithms for Minimum Cost Flow Problems, Second Edition*. CRC Press, New York, 2015.
- G. Ruhe. *Algorithmic Aspects of Flows in Networks*. Kluwer Academic Publishers, Dordrecht, The Netherlands, 1991.
- H. Zheng, Y.-C. Chiu, and P. B. Mirchandani. On the system optimum dynamic traffic assignment and earliest arrival flow problems. *Transportation Science*, 49(1):13–27, 2015. doi: <http://dx.doi.org/10.1287/trsc.2013.0485>.